collabolatte - Epic Breakdown

Overview

This document provides the complete epic and story breakdown for collabolatte, decomposing the requirements from the PRD, UX Design if it exists, and Architecture requirements into implementable stories.

Requirements Inventory

Functional Requirements

FR1: Users authenticate via Azure Entra ID (EasyAuth); no custom credential storage FR2: System captures user email and display name from Entra ID claims on first login FR3: Users can optionally provide department and location if not available from directory FR4: Users can view their own participation history (programmes joined, past matches) FR5: Users can view programmes they are eligible to join FR6: Users can join a programme with a single action FR7: Users can leave a programme at any time; if mid-cycle, pending match is cancelled and match partner is notified FR8: Users can view programmes they are currently enrolled in FR9: Users can pause participation in a programme (skip next cycle) FR10: Users receive confirmation of programme join/leave actions FR11: System executes matching algorithm on a predictable schedule (weekly or monthly; default monthly; configured as an ops setting, not user-configurable) FR12: All programme participants are eligible for matching; organisational boundary filtering is a Growth-phase capability (MVP is random-only) FR13: Random matching with architecture supporting future algorithm configurability FR14: Matching algorithm avoids repeat pairings within configurable window (default: 3 cycles); historical matches stored per programme FR15: If participant count is odd, one participant is gracefully excluded with explanation FR16: Match notification email contains: matched participant name, Teams deep link, and a simple first-move prompt with suggested intro copy (copy-paste friendly) FR17: Email is sent via Azure Communication Services to user's Entra ID email FR18: Teams deep link opens 1:1 chat with matched participant FR19 (Deferred / Post-MVP): Conversation starter selection (post-MVP conversation support) FR20 (Deferred / Post-MVP): Reminder per match cycle (not in MVP; risks obligation) FR21: Programme Owner can create a new programme with name, description, and cadence FR22: Programme Owner can invite participants by email address or CSV upload FR23: Programme Owner can view aggregate participation metrics (joined, active, paused) [Growth] FR24: Matching cadence is configurable as an ops setting (weekly or monthly; default monthly); no Programme Owner UI for cadence in MVP FR25: Programme Owner can edit programme name and description (cadence is ops-configured in MVP) FR26: Admin can view all programmes in the system FR27: Admin can deactivate a programme FR28: Admin can assign Programme Owner role to users FR29: Admin cannot view individual participation or match data FR29a: Admin can view ops-only programme health (cycle execution, notification delivery health, trust guardrails status, operational errors) gated behind minimum-N ≥5 FR30: No individual-level API endpoints exist for admin/owner queries about specific participants; individual participants can access their own data via authenticated endpoints FR31: No tracking of whether matched participants actually met FR32: All personal data deletable upon user request (GDPR Article 17) FR33: No analytics on message content or conversation outcomes FR34: System logs events, not behaviours (see Event Logging) FR35: System logs programme lifecycle events (created, paused, deactivated) FR36: System logs matching execution events (run started, completed, participants matched) FR37: Privacy-preserving aggregate queries only; minimum 5 participants required for any aggregate query FR38: Event log entries are immutable and timestamped FR39: No individual-level event queries exposed to admin/owner roles FR41: Users can view a "What we collect" transparency page explaining data handling

FR42: Programme Owner receives notification when matching cycle completes, including status and any system errors (no participation counts)

NonFunctional Requirements

NFR1: Page loads complete within 3 seconds on standard corporate network NFR2: No hard real-time requirements NFR3: System remains usable under degraded conditions NFR4: All data encrypted in transit (TLS 1.2+) NFR5: All data encrypted at rest NFR6: Authentication via Azure Entra ID only; no custom credential storage NFR7: Least-privilege access: users see only their own data; owners see only aggregate NFR8: Clear data retention policy: personal data deleted within 30 days of account deletion request NFR9: No dark telemetry or behavioural tracking NFR10: Tenant isolation: single-tenant MVP with architecture supporting future multi-tenant separation NFR11: Best-effort availability; no formal SLA for MVP NFR12: One hour of downtime during business hours is acceptable NFR13: Clear failure behaviour: no silent errors, no partial matches NFR14: Matching algorithm is atomic: runs completely or not at all NFR15: Match notifications sent within 60 minutes of algorithm completion NFR16: Email delivery includes retry logic for transient failures NFR17: Failed email delivery logged for operational review NFR18: WCAG 2.1 AA as aspiration, not hard gate NFR19: Keyboard navigation for all core flows NFR20: Readable contrast ratios (4.5:1 minimum for text) NFR21: Screen-reader friendliness where straightforward NFR22: Architecture supports future multi-tenant expansion NFR23: No requirement to prove scale in MVP NFR24: Consumption-based infrastructure (Azure Functions, Table Storage) NFR25: Sufficient logging to debug failures and explain what happened NFR26: No user-level behavioural analytics beyond operational necessity NFR27: Application Insights or equivalent for error monitoring

Additional Requirements

- Starter template: monorepo with Vite React + TypeScript SPA, Azure Functions (.NET isolated), and 11ty marketing site; pnpm workspaces; two Azure Static Web Apps (app+API and marketing).
- Authentication via Azure Entra ID EasyAuth; no custom auth flows; claims treated as untrusted per request.
- Authorisation is role-based (Participant, Programme Owner, Admin) with a small allowlist stored in Table Storage; no manager visibility.
- Data storage is Azure Table Storage with programme-scoped partitioning; MatchId deterministic; idempotent matching keyed by ProgrammeId + CycleDate.
- API: REST JSON, plural noun routes; Problem Details (RFC 9457) for errors; OpenAPI.NET specs aligned to DTOs and contract tests.
- Logging: immutable system-event logs only (no behaviour analytics); correlation IDs are GUIDs.
- Retention: match records retained 12 months; deletion on request; aggregate reporting minimum N = 5.
- Deployment: SWA config via staticwebapp.config.json; no Key Vault or App Insights in MVP (upgrade hooks only).
- Web-only MVP; primary interactions via email and Teams links; notifications are the core experience.
- Single-screen, low-pressure UI; minimal navigation; one clear primary action with safe secondary actions (Skip/Later/Leave).
- No reminders, no feedback collection, no dashboards, no gamification, no profile optimisation.
- Transparency surfaces: short "What we store" link on join and match screens; calm, plain-language copy.
- Responsive design: desktop-first but mobile-safe; breakpoints at 360px, 768px, 1024px, 1280px.

- Accessibility: WCAG 2.1 AA aspiration; visible focus states; keyboard navigation; readable contrast; respect reduced-motion.

FR Coverage Map

FR1: Epic 1 - Join & Trust FR2: Epic 1 - Join & Trust FR3: Epic 1 - Join & Trust FR4: Epic 4 - Stay Opted-In FR5: Epic 1 - Join & Trust FR6: Epic 1 - Join & Trust FR7: Epic 4 - Stay Opted-In FR8: Epic 4 - Stay Opted-In FR9: Epic 4 - Stay Opted-In FR10: Epic 4 - Stay Opted-In FR11: Epic 2 - Get Matched FR12: Epic 2 - Get Matched FR13: Epic 2 - Get Matched FR14: Epic 2 - Get Matched FR15: Epic 2 - Get Matched FR16: Epic 3 - Have the Conversation FR17: Epic 3 - Have the Conversation FR18: Epic 3 - Have the Conversation FR19: Deferred / Post-MVP (not in epics) FR20: Deferred / Post-MVP (not in epics) FR21: Epic 5 - Operate Safely FR22: Epic 5 - Operate Safely FR23: Epic 5 - Operate Safely FR24: Epic 5 - Operate Safely FR25: Epic 5 - Operate Safely FR26: Epic 5 - Operate Safely FR27: Epic 5 - Operate Safely FR28: Epic 5 - Operate Safely FR29: Epic 5 - Operate Safely FR29a: Epic 5 - Operate Safely (Growth) FR30: Epic 5 - Operate Safely FR31: Epic 5 - Operate Safely FR32: Epic 5 - Operate Safely FR33: Epic 5 - Operate Safely FR34: Epic 5 - Operate Safely FR35: Epic 5 - Operate Safely FR36: Epic 5 - Operate Safely FR37: Epic 5 - Operate Safely FR38: Epic 5 - Operate Safely FR39: Epic 5 - Operate Safely FR41: Epic 1 - Join & Trust FR42: Epic 5 - Operate Safely

Epic List

Epic 1: Join & Trust

Participants can safely opt in, understand what is happening, and trust the system from the first touch. **FRs covered:** FR1, FR2, FR3, FR5, FR6, FR41

Epic 2: Get Matched

Participants reliably receive predictable, random matches on the agreed cadence. **FRs covered:** FR11, FR12, FR13, FR14, FR15

Epic 3: Have the Conversation

Participants get a low-effort, low-pressure invitation that makes the first move easy. **FRs covered:** FR16, FR17, FR18

Epic 4: Stay Opted-In

Participants can pause, leave, and return without penalty, and see their own history. **FRs covered:** FR4, FR7, FR8, FR9, FR10

Epic 5: Operate Safely

Programme Owners and Admins can run the system safely with strict trust guardrails and operational controls. **FRs covered:** FR21, FR22, FR23, FR24, FR25, FR26, FR27, FR28, FR29, FR29a (Growth), FR30, FR31, FR32, FR33, FR34, FR35, FR36, FR37, FR38, FR39, FR42

Deferred / Post-MVP (Not in Epics)

- FR19: Conversation starter selection (post-MVP conversation support)
- FR20: Reminder per match cycle (not in MVP; risks obligation)

Rule: these items remain explicitly out of MVP scope unless re-approved; they must not appear implicitly as “small enhancements” because they can erode the anti-mandate trust contract.

Trust Copy Checklist (Single Source of Truth)

Applies to: join screen, match screen, pause/leave confirmations, invites, cycle notices, transparency pages, and match notification emails.

Required statements (where relevant):

- Participation is optional and consequence-free.
- No behavioural tracking; no manager visibility.
- What we store / what we do not store is explained in plain language.

Forbidden patterns (must not appear):

- Any “obligation” framing: “required”, “must”, “expected”, “mandatory”, “compliance”, “attendance”.
- Any surveillance framing: “tracked”, “monitoring”, “engagement”, “adoption”, “participation rate”, “performance”.
- Any gamification framing: “streak”, “leaderboard”, “score”, “points”.

Tone rules (testable):

- Neutral, non-celebratory, non-judgemental language.
- Never guilt users for skipping/ignoring a match.
- Failure states include a calm explanation and do not introduce urgency.

Implementation Sequencing (Reasoning via Planning)

Phase 0 - Foundations (Story 1.0)

- Scaffold repo + SWA config + auth wiring + storage placeholders
- Outcome: devs can run web/api locally and deploy a thin skeleton

Phase 1 - Join & Trust (Stories 1.1-1.6)

- First contact trust surface
- Entra auth flow
- What-we-store transparency
- One-click join
- Leave/pause clarity at join
- Default participant role
- Outcome: users can safely opt in and trust the system

Phase 2 - Get Matched (Stories 2.1-2.7)

- Inclusion in cycle
- Cadence visibility
- Matching logic (random)
- Repeat avoidance + odd handling with calm notice
- Future algorithm configurability

- Non-participation handled silently
- Failures do not leak
- Outcome: the ritual runs without user effort

Phase 3 - Have the Conversation (Stories 3.1-3.4)

- Essential match email
- First-move prompt
- Teams deep link
- Silence acceptable
- Outcome: low-pressure invite delivered

Phase 4 - Stay Opted-In (Stories 4.1-4.5)

- Participation status
- Pause next cycle
- Leave mid-cycle handling
- Calm confirmations
- Minimal history
- Outcome: voluntary loop stays safe

Phase 5 - Operate Safely (Stories 5.1-5.8)

- Single programme config
- Invites
- Deactivate
- Role assignment
- Aggregate-only visibility
- Cycle completion notice
- Privacy controls
- System-event logs
- Outcome: hands-off operations with trust guardrails

Epic Failure Modes & Guardrails

Epic 1: Join & Trust

Failure modes:

- Trust undermined by silently capturing extra fields or implying monitoring
- Join flow feels like onboarding (too long, too many steps) Mitigations:
- Strictly limit data capture to name/email/department/location
- Single-screen join with explicit "What we store" link
- No multi-programme UI in MVP

Epic 2: Get Matched

Failure modes:

- Matching becomes algorithm feature creep (preferences, optimisation)

- Inconsistent cadence erodes predictability Mitigations:
- Random + repeat-avoidance only; no preferences or boundary filters in MVP
- Match runs idempotent and logged; cadence communicated

Epic 3: Have the Conversation

Failure modes:

- Notification UX drifts into reminders or nudges (pressure)
- Tracking whether meetings happened violates trust Mitigations:
- One notification per match; no reminders
- No meeting-happened tracking or click analytics

Epic 4: Stay Opted-In

Failure modes:

- Pause/leave becomes a status signal
- Re-engagement nudges create pressure Mitigations:
- Pause/leave affects matching eligibility only; no behaviour prompts
- No re-engagement campaigns

Epic 5: Operate Safely

Failure modes:

- Admin tooling becomes a dashboard (metrics pressure)
- Individual visibility creeps in Mitigations:
- Aggregate-only, minimum-N enforced; no individual-level endpoints
- FR29a is MVP but strictly ops-only; no participation counts or adoption signals
- No multi-programme UI in MVP

Pre-mortem Risk Summary

Failure scenario (6 months later): Adoption fizzled, matches became irregular, trust wobbled, and participation quietly declined.

Likely causes and prevention (by epic):

Epic 1: Join & Trust

- Cause: Join felt like HR compliance; transparency link buried or too legalistic
- Prevention: Keep join copy human and explicit; surface "What we store" in the primary flow

Epic 2: Get Matched

- Cause: Cadence drifted; repeat matches happened too soon
- Prevention: Hard-log cycle runs with visible cadence; enforce repeat-avoidance window

Epic 3: Have the Conversation

- Cause: Notification copy too generic; Teams link unreliable
- Prevention: Treat notification copy as core UX; validate Teams link generation

Epic 4: Stay Opted-In

- Cause: Pause hidden; return felt cold
- Prevention: Make pause/leave explicit; include "welcome back" tone without reminders

Epic 5: Operate Safely

- Cause: Metrics pressure led to dashboard creep; small cohorts exposed individuals
- Prevention: Enforce minimum-N; keep ops to system health + controls only

Stakeholder Round Table Summary

PM (Outcome clarity):

- Epic 1 delivers value when opt-in is safe and obvious; avoid extra fields
- Epic 2 must stay "random + repeat-avoidance" to remain MVP
- Epic 5 must remain ops-only; keep FR29a Growth visible

Architect (Dependency hygiene):

- Epic 3 depends on Epic 2; keep it strictly downstream of match records
- Epic 4 should not introduce analytics tables or metrics surfaces
- Keep data writes scoped per epic to avoid coupling

UX (Trust-first experience):

- "Have the Conversation" must feel like a gentle invite, not a push
- "Stay Opted-In" should be one-click with calm microcopy
- Avoid multi-programme UI elements in MVP screens

Socratic Checkpoints (Round 2)

1. If Epic 4 never shipped, would the system still feel voluntary?
   - It would feel less safe; pause/leave is essential to voluntariness.
2. Is "Have the Conversation" its own outcome or just a notification detail?
   - It is its own outcome; invitation design directly affects action.
3. What happens if admins demand "just a small dashboard"?
   - Trust posture changes; treat as Growth, not MVP.
4. What is the minimal truthful promise in the join copy?
   - "Opt-in, no tracking, no consequences if you ignore."
5. What is the smallest irreversible harm?
   - Any exposure of individual participation or match history to hierarchy.

Epic 1: Join & Trust

Participants can safely opt in, understand what is happening, and trust the system from the first touch. **FRs covered:** FR1, FR2, FR3, FR5, FR6, FR41 **NFR Focus:** NFR1 (baseline performance), NFR6 (Entra ID only), NFR18-21 (accessibility basics).

Epic 1 Prerequisites: Minimal project scaffolding and Azure foundations are in place (repo structure, SWA config stubs, Entra ID auth wiring baseline, and storage connection placeholders).

Azure Setup Note (pre-Story 1.0):

- Create Azure Static Web Apps resources (app+api, marketing)
- Configure Entra ID app registration and EasyAuth for SWA
- Create Storage Account (Table Storage)
- Create Azure Communication Services resource (email)
- Set required app settings and secrets in SWA

Story 1.0: Document infrastructure provisioning (Bicep)

As a delivery team, I want a documented Bicep-based infrastructure plan in /infra, So that we can provision Azure consistently when we are ready.

Acceptance Criteria:

Given infrastructure is not yet provisioned, **When** the /infra documentation is created, **Then** it defines the resources required for MVP (2x SWA, Storage Account, ACS), **And** it specifies naming conventions per Azure resource rules, **And** it lists required parameters (project, environment, region, identifier), **And** it documents Entra ID and EasyAuth setup steps.

Given the document exists, **When** a developer reviews /infra/README.md, **Then** they can follow it to implement Bicep later without ambiguity.

Story 1.1: Set up initial project from starter template

As a delivery team, I want to set up the initial project from the approved starter templates, So that Join & Trust stories can be implemented without blocking setup work.

Acceptance Criteria:

Given the starter templates are approved in Architecture, **When** we initialise the project, **Then** the repo is scaffolded as a monorepo with `/apps/web`, `/apps/api`, and `/apps/marketing`, **And** the web app is created from the official Vite React TypeScript template, **And** the API is initialised with Azure Functions (.NET isolated), **And** the marketing site is initialised with 11ty.

Given the scaffolding exists, **When** baseline configuration is applied, **Then** `staticwebapp.config.json` and SWA workflow stubs exist for app+api and marketing, **And** Entra ID auth wiring is configured at the platform level (no custom auth code), **And** storage connection placeholders are defined for local and cloud environments.

Given the foundations are in place, **When** a developer starts work on Epic 1 stories, **Then** they can run the web and API projects locally without additional scaffolding.

Story 1.2: First contact feels safe

As a prospective participant, I want to understand what Collabolatte is and is not before doing anything, So that I can decide whether it feels safe and optional.

Acceptance Criteria:

Given I land on the Collabolatte app unauthenticated, **When** I view the first screen, **Then** I can immediately see that participation is optional and lightweight, **And** the screen explicitly states that behaviour is not tracked, **And** the copy conforms to the Trust Copy Checklist, **And** no action is required to understand this.

Given I have not signed in, **When** I read the first screen, **Then** the trust message is in plain English without legal language, **And** it conforms to the Trust Copy Checklist.

Story 1.3: Authenticate without friction

As a prospective participant, I want to authenticate using my corporate Entra ID without creating an account, So that joining feels invisible and low-effort.

Acceptance Criteria:

Given I choose to sign in, **When** I authenticate via Entra ID, **Then** I am signed in without creating any custom credentials, **And** I do not see any additional create account steps.

Given I have authenticated successfully, **When** I land back in the app, **Then** I see clear reassurance that only basic identity data is used.

Story 1.4: See what we store (and do not)

As a prospective participant, I want to see a clear, plain-English summary of what data is stored about me before joining, So that I can decide with confidence.

Acceptance Criteria:

Given I am on the join screen, **When** I look for data handling information, **Then** I can see a short, plain-English summary of what is stored and what is not, **And** it conforms to the Trust Copy Checklist.

Given I want more detail, **When** I select the "What we store" link, **Then** I can view the full explanation without legalese, **And** I can return to the join screen without losing my place.

Story 1.5: Join with a single action

As a prospective participant, I want to opt in with one clear, reversible action, So that joining feels low-stakes.

Acceptance Criteria:

Given I am on the join screen, **When** I choose to join, **Then** there is a single primary Join action, **And** I am not asked for a bio, interests, or extra details, **And** the language avoids commitment or obligation, **And** it conforms to the Trust Copy Checklist.

Given I select Join, **When** the action completes, **Then** I see a clear confirmation that I am now opted in.

Story 1.6: Know I can leave or pause

As a prospective participant, I want to know I can pause or leave at any time, So that I feel safe opting in.

Acceptance Criteria:

Given I am on the join screen, **When** I read the opt-in information, **Then** I can clearly see that leaving or pausing is allowed at any time, **And** it is stated as consequence-free.

Given I have not joined yet, **When** I read the join copy, **Then** pausing or leaving is mentioned explicitly at join time.

Story 1.7: Be treated as a participant by default

As a newly joined participant, I want to be treated as a standard participant with no special visibility or responsibilities, So that I do not feel pressure or role confusion.

Acceptance Criteria:

Given I have joined, **When** I view the post-join state, **Then** I do not see admin or programme-owner surfaces, **And** I do not see metrics or status indicators beyond "You may be matched."

Given I am a standard participant, **When** I use the app, **Then** my default role is participant only.

Epic 2: Get Matched

Participants reliably receive predictable, random matches on the agreed cadence. **FRs covered:** FR11, FR12, FR13, FR14, FR15 **NFR Focus:** NFR2 (no real-time), NFR13-14 (clear failure behavior, atomic matching).

Story 2.1: Included in the next matching cycle

As a joined participant, I want to be automatically included in the next matching cycle, So that I do not have to do anything after joining.

Acceptance Criteria:

Given I have joined the programme, **When** a matching cycle runs, **Then** I am included by default unless I have paused or left.

Given I am a joined participant, **When** I view my status, **Then** I do not need to confirm availability, **And** I am not asked for preferences.

Story 2.2: Matching runs on a predictable cadence

As a joined participant, I want matching to run on a clear, predictable cadence, So that I know when to expect my next match.

Acceptance Criteria:

Given I am a joined participant, **When** I view the matching cadence, **Then** I can see it in simple terms (e.g., monthly by default), **And** I can see when the next match is expected.

Given a matching run is delayed or fails, **When** I view the participant experience, **Then** I do not see error messaging or urgency, **And** the experience remains calm and unchanged.

Story 2.3: Matched with real people

As a joined participant, I want to be matched with other real participants, So that the programme actually creates connections.

Acceptance Criteria:

Given a matching cycle runs, **When** I am included, **Then** I am matched into a group with at least one other participant.

Given a matching cycle runs, **When** matches are generated, **Then** the system pairs eligible participants into matches without manual intervention.

Story 2.4: Avoid repeats and handle odd counts

As a joined participant, I want matching to avoid obvious repeats and handle odd numbers gracefully, So that the experience feels fair and reliable.

Acceptance Criteria:

Given a matching cycle runs, **When** matches are generated, **Then** recent pairings within the repeat-avoidance window are not repeated.

Given a matching cycle runs with an odd number of eligible participants, **When** matches are generated, **Then** one participant is gracefully excluded for that cycle, **And** they receive a calm notification explaining they will be included next cycle.

Story 2.5: Random now, configurable later

As a programme sponsor, I want matching to be random in MVP but architected for future configurability, So that different programmes can later adopt light matching rules without rework.

Acceptance Criteria:

Given MVP matching runs, **When** matches are generated, **Then** the algorithm is random and neutral (no optimisation or scoring).

Given the system is designed for MVP, **When** matching is implemented, **Then** the architecture allows future algorithm configurability without changing participant expectations.

Story 2.6: Non-participation is handled silently

As a participant, I want non-participation to be handled without awkward messaging, So that I do not feel pressure.

Acceptance Criteria:

Given I am paused or ineligible for a cycle, **When** a matching run occurs, **Then** I receive no 'you were skipped' messaging.

Given I am paused or ineligible, **When** the next cycle runs, **Then** I am included again automatically once eligible.

Story 2.7: Matching failure does not leak to users

As a participant, I want internal matching failures to remain invisible to me, So that trust in the system is preserved.

Acceptance Criteria:

Given a matching run fails internally, **When** I view the participant experience, **Then** I see no error messages or partial notifications, **And** any participant-visible messaging conforms to the Trust Copy Checklist.

Given a matching run fails, **When** the system recovers for the next cycle, **Then** the participant experience remains calm and unchanged.

Epic 3: Have the Conversation

Participants get a low-effort, low-pressure invitation that makes the first move easy. **FRs covered:** FR16, FR17, FR18 **NFR Focus:** NFR15-17 (email delivery reliability).

Story 3.1: Match notification contains the essentials

As a matched participant, I want a single calm match email with everything I need to take the first step, So that I can decide whether to engage without friction.

Acceptance Criteria:

Given a match is created for me, **When** the notification email is sent, **Then** it includes the names of matched participants, **And** it includes the Teams deep link(s), **And** it states that the conversation is optional, **And** it conforms to the Trust Copy Checklist.

Given I receive the match email, **When** I view it, **Then** it contains no tracking pixels, read receipts, or reminders implied, **And** it conforms to the Trust Copy Checklist.

Story 3.2: First-move prompt reduces social friction

As a matched participant, I want a calm, copy-pasteable first-move prompt, So that starting the conversation feels easy and non-awkward.

Acceptance Criteria:

Given I receive a match email, **When** I read the prompt, **Then** the tone is calm and optional, **And** the prompt can be copy-pasted without editing, **And** it explicitly permits ignoring or adapting the prompt, **And** it conforms to the Trust Copy Checklist.

Story 3.3: Teams deep link works reliably

As a matched participant, I want the Teams link to open the correct chat reliably, So that I can start the conversation without confusion.

Acceptance Criteria:

Given I click the Teams deep link from the match email, **When** Teams is available, **Then** it opens a 1:1 chat with the matched participant.

Given Teams is not available on my device, **When** I click the link, **Then** I receive a graceful fallback (no error wall), **And** I am not forced into calendar scheduling.

Story 3.4: Silence is an acceptable outcome

As a participant, I want no negative consequences if I do nothing after a match email, So that I never feel pressured to engage.

Acceptance Criteria:

Given I receive a match email, **When** I take no action, **Then** I receive no follow-ups or warnings, **And** no new state is shown that implies failure.

Given I take no action, **When** the next cycle runs, **Then** the system proceeds normally without calling out my inactivity.

Epic 4: Stay Opted-In

Participants can pause, leave, and return without penalty, and see their own history. **FRs covered:** FR4, FR7, FR8, FR9, FR10 **NFR Focus:** NFR7 (least-privilege access), NFR9 (no dark telemetry).

Epic 4 Clarification: Ignoring a match is not treated as a pause or leave. Non-response does not change participation state.

Epic 4 Anti-stories:

- No are you sure friction
- No warnings about missing opportunities
- No manager visibility into status
- No historical analytics or trend views
- No re-engagement nudges

Story 4.1: See my participation status

As a participant, I want to clearly see whether I am opted in, paused, or left, So that I understand my current state without pressure.

Acceptance Criteria:

Given I am a participant, **When** I view my participation status, **Then** the status is factual and neutral, **And** it uses no judgmental language, **And** it includes no prompts to change state.

Story 4.2: Pause next cycle

As a participant, I want to pause participation for the next cycle with one action, So that I can step back temporarily without friction.

Acceptance Criteria:

Given I am opted in, **When** I choose to pause, **Then** the pause applies to the next matching cycle only, **And** it expires after one cycle unless I pause again, **And** no explanation is required.

Given I pause, **When** the action completes, **Then** I see a calm acknowledgement with no extra ceremony.

Story 4.3: Leave at any time

As a participant, I want to leave the programme at any time with one action, So that opting out feels as safe as opting in.

Acceptance Criteria:

Given I am opted in, **When** I choose to leave, **Then** I leave with one action and no friction.

Given I leave mid-cycle, **When** the leave is processed, **Then** any pending match is cancelled, **And** other participants are not notified.

Given I leave, **When** I see the confirmation, **Then** it reassures me there are no consequences.

Story 4.4: Calm confirmation of state changes

As a participant, I want a quiet confirmation when I join, pause, or leave, So that I am not left uncertain about what happened.

Acceptance Criteria:

Given I change my participation state (join, pause, leave), **When** the action completes, **Then** I see an informational confirmation, **And** it is inline (no email required), **And** the language reinforces reversibility.

Story 4.5: View my own participation history

As a participant, I want a minimal record of my own participation history, So that I can remember my involvement without performance pressure.

Acceptance Criteria:

Given I view my participation history, **When** it is shown, **Then** it includes my join date, pause or leave actions, and past match dates and names only.

Given I view my history, **When** I review it, **Then** it does not include counts, streaks, or engagement indicators.

Epic 5: Operate Safely

Programme Owners and Admins can run the system safely with strict trust guardrails and operational controls. **FRs covered:** FR21, FR22, FR23, FR24, FR25, FR26, FR27, FR28, FR29, FR29a (Growth), FR30, FR31, FR32, FR33, FR34, FR35, FR36, FR37, FR38, FR39, FR42 **NFR Focus:** NFR4-10 (security/privacy/retention), NFR25-27 (logging and monitoring).

Epic 5 Clarification: Programme Owners are enablers, not operators. They cannot intervene in individual matches, view individual participation, or influence outcomes.

Epic 5 Anti-stories:

- No individual-level reporting
- No engagement scoring
- No nudges or reminders triggered by metrics

- No manager visibility
- No optimisation controls for matching

Story 5.1: Programme exists and can be configured

As a Programme Owner, I want to set the programme name and description and see the programme cadence, So that the programme is clear and runs predictably without expanding scope.

Acceptance Criteria:

Given a single programme exists, **When** I set or update its name or description, **Then** the changes apply to future cycles only, **And** the MVP remains limited to a single programme.

Given I am a Programme Owner, **When** I view the programme settings, **Then** I can see the cadence in simple terms (monthly by default; weekly optional via ops), **And** I cannot change cadence via any UI surface in MVP, **And** participant-facing copy about cadence conforms to the Trust Copy Checklist.

Story 5.2: Participants can be invited

As a Programme Owner, I want to invite participants by email or CSV, So that I can open the programme without friction.

Acceptance Criteria:

Given I am a Programme Owner, **When** I send invites by email or CSV, **Then** the invite message states participation is optional and low-pressure, **And** it conforms to the Trust Copy Checklist.

Given invites are sent, **When** participants join, **Then** the system does not track acceptance beyond join status.

Story 5.3: Programme lifecycle controls

As an Admin, I want to deactivate the programme safely, So that matching can be stopped without disrupting trust.

Acceptance Criteria:

Given the programme is active, **When** I deactivate it, **Then** future matching stops, **And** historical data is not deleted immediately, **And** participants are not notified loudly.

Story 5.4: Role assignment is possible

As an Admin, I want to assign Programme Owner roles from a controlled allowlist, So that responsibility is explicit and auditable.

Acceptance Criteria:

Given I am an Admin, **When** I assign a Programme Owner role, **Then** the allowlist is stored in Table Storage, **And** there is no self-service role escalation, **And** the change is auditable.

Story 5.5: Aggregate-only visibility

As a Programme Owner or Admin, I want aggregate, non-identifying visibility only, So that trust is preserved.

Acceptance Criteria:

Given I view participation information, **When** aggregates are shown or exported, **Then** minimum-N thresholds are enforced everywhere.

Given aggregates are shown, **When** I view them, **Then** they include cycle execution health, notification delivery health, trust guardrails status, and operational errors (with correlation IDs) only, **And** they exclude participation counts, adoption signals, per-cycle performance views, or success metrics.

And all ops-only health is gated behind minimum-N ≥5.

Story 5.6: Matching cycle completion notice

As a Programme Owner, I want a calm operational notice when a cycle completes, So that I know the system is running without pressure.

Acceptance Criteria:

Given a matching cycle completes, **When** the notice is sent, **Then** it states the cycle completed, **And** it includes any system errors if relevant, **And** it does not include participation stats or celebratory language, **And** it conforms to the Trust Copy Checklist.

Story 5.7: Privacy controls are real

As a participant, I want deletion requests honoured and no behavioural tracking, So that the privacy promise is enforced.

Acceptance Criteria:

Given I request deletion, **When** the request is processed, **Then** identifying data is removed within the stated window, **And** only minimal system audit data is retained.

Given the system operates, **When** events are logged, **Then** no behavioural or engagement tracking is recorded.

Story 5.8: System-event logging

As an Admin, I want immutable system-event logs for audit and debugging, So that operations are transparent without surveillance.

Acceptance Criteria:

Given system events occur, **When** they are logged, **Then** logs include programme create/update/deactivate, matching run start/complete/fail, and notification attempt/sent/fail only, **And** they exclude user-level behaviour events, **And** correlation IDs are included.